WO0039947

Publication Title:

BROADCAST DATA ACCESS SYSTEM FOR MULTIMEDIA CLIENTS IN A BROADCAST NETWORK ARCHITECTURE

Abstract:

A broadcast data access system is provided for receiving broadcast data by applications residing on a multimedia client, where the broadcast data is a set of modules on a data carousel that are being broadcast over a broadcast network. The broadcast data access system includes an interest manager configured to store a plurality of interests, such that each interest identifies an available module on the data carousel being requested by an application. The system further includes at least one application having registered an interest for a first module with the interest manager, and a dispatcher distributing the first module to the requesting application by accessing the interest manager

---

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : H04H 1/00 | A2

(11) International Publication Number: WO 00/39947

(43) International Publication Date: 6 July 2000 (06.07.00)

(21) International Application Number: PCT/US99/30249

(22) International Filing Date: 17 December 1999 (17.12.99)

(30) Priority Data:
09/219,714 23 December 1998 (23.12.98) US

(71) Applicant: POWERTV, INC. [US/US]; Suite 100, 20833 Steven Creek Blvd., Cupertino, CA 95014-2154 (US).

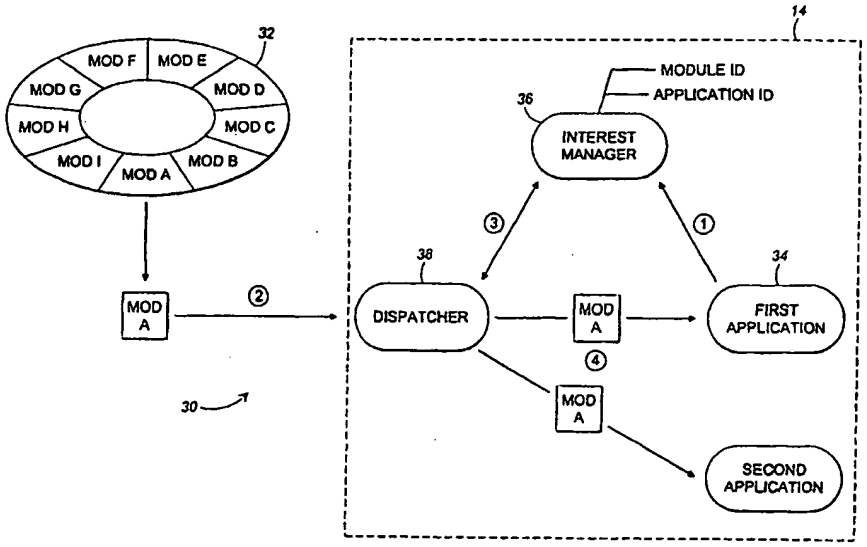
(72) Inventor: STALKER, Altan, J.; 6315 Mayo Drive, San Jose, CA 95123 (US).

(74) Agents: MASSARONI, Kenneth, M. et al.; Scientific-Atlanta, Inc., Intellectual Property Department, One Technology Parkway South, Norcross, GA 30092 (US).

(81) Designated States: KR, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: A BROADCAST DATA ACCESS SYSTEM FOR MULTIMEDIA CLIENTS IN A BROADCAST NETWORK ARCHITECTURE



(57) Abstract

A broadcast data access system is provided for receiving broadcast data by applications residing on a multimedia client, where the broadcast data is a set of modules on a data carousel that are being broadcast over a broadcast network. The broadcast data access system includes an interest manager configured to store a plurality of interests, such that each interest identifies an available module on the data carousel being requested by an application. The system further includes at least one application having registered an interest for a first module with the interest manager, and a dispatcher distributing the first module to the requesting application by accessing the interest manager.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

A BROADCAST DATA ACCESS SYSTEM FOR MULTIMEDIA CLIENTS IN A BROADCAST NETWORK ARCHITECTURE

BACKGROUND OF THE INVENTION

The present invention relates generally to a broadcast data access system, and more particularly to an architecture for supporting applications that receive broadcast data from a data carousel over a broadcast network.

In a broadcast network architecture, various types of data can be delivered from a server to a group of multimedia clients. Typically, multimedia clients do not have enough resources to store all of the data that is being broadcast over the network. Even if the client could store all of the data, there is no guarantee that the client will receive an error-free copy of the data in a single transmission of the data. Moreover, the client has no way of requesting that a server resend missing or defective data. Since the data is being sent to many clients, it might also be prohibitive to resend missing or defective data to each of the clients which request it.

A broadcast data carousel is commonly used for transporting data in a broadcast environment. This underlying mechanism for transporting data is defined in the MPEG-2 DSM-CC specification (i.e., ISC/IEC 13818-6). Using this mechanism, the server repeatedly sends data over a period of time so that a client which is interested in the data may receive it only when it is required. If a client misses some of the data or receives defective data, it waits for the next broadcast of the data to receive any data that it may need.

A Broadcast File System (BFS) provides a layer on top of the broadcast data carousel that hides the details of the underlying transport mechanism from the server and

clients. In particular, BFS creates a mapping between the carousel/file number and a module name. As a result, the server and clients view the broadcast data as a standard hierarchical file system similar to files found on a disk operating system.

Therefore, it would be desirable to provide a broadcast data access system for receiving broadcast data from a data carousel in a simple, efficient, and flexible manner. It should support multiple data sources between the broadcast network and the multimedia client, such that each source can receive a different form of encoded broadcast data. In addition, the broadcast data access system should be able to efficiently process data packets received in a non-sequential order, as well as simultaneously fulfill multiple requests for the same data packets by different applications. To lessen processing overhead, filters are dynamically installed on the client. Lastly, the present invention should provide a method for downloading and synchronizing a directory module with the content of the data carousel being broadcast to the client.

SUMMARY OF THE INVENTION

In accordance with the teachings of the present invention, a broadcast data access system is provided for receiving broadcast data by applications residing on a multimedia client, where the broadcast data is a set of modules on a data carousel that are being broadcast over a broadcast network. The broadcast data access system includes an interest manager configured to store a plurality of interests, such that each interest identifies an available module on the data carousel being requested by an application. The system further includes at least one application having registered an interest for a first module with the interest manager, and a dispatcher distributing the first module to the requesting application

by accessing the interest manager.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a diagram depicting the basic components in a typical broadcast file system in accordance with the present invention;

Figure 2 is a data flow diagram illustrating a preferred embodiment of the broadcast data access system of the present invention;

Figure 3 is a block diagram showing the software architecture for the broadcast data access system of the present invention which resides on each multimedia client;

Figure 4 is a diagram showing dynamic filter installation in accordance with the present invention;

Figures 5A and 5B are a detailed flowchart illustrating a preferred embodiment of the data block processor of the present invention; and

Figure 6 is a detailed flowchart illustrating a preferred embodiment of a portion of the data block processor that handles downloading and synchronizing the BFS directory onto the client.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

The following description of the present invention is merely exemplary in nature and is in no way intended to limit the invention or its uses. Moreover, the following description, while depicting a broadcast data access system that is designed to reside on a conventional set top box, is intended to adequately teach one skilled in the art to make and use a similar architecture for a variety of consumer multimedia clients including, but not

limited to, intelligent televisions, Internet terminals and advanced DVD players.

A Broadcast File System 10 (BFS), as depicted in Figure 1, provides an architecture for delivering various types of data from a BFS server 12 (or group of servers) to a plurality of multimedia clients 14 over a broadcast network 16. It is also envisioned that a two-way communication network could be used in conjunction with the broadcast file system 10 of the present invention. The underlying mechanism for transporting data across the network 16 relies on a broadcast data carousel which is defined in the MPEG-2 DSM-CC specification (i.e., ISC/IEC 13818-6). Typically, broadcast data is grouped into files that are subdivided into fixed-size data blocks and then broadcast in a non-sequential order using the data carousel mechanism. However, BFS 10 of the present invention provides a layer on top of the broadcast data carousel that hides the details of this underlying transport mechanism from the server 12 and clients 14. Within a data carousel, individual data files are called modules. Since modules are identified by numbers (not names), BFS 10 creates a mapping between file numbers and module names. In this way, the server 12 and clients 14 of BFS 10 view these modules in a standard hierarchical file system similar to files found on a disk operating system.

BFS server 12 is the component responsible for storing, assembling and delivering modules across the network 16. While the following discussion is provided with reference to one data carousel, it is readily understood that the explanation is applicable to more than one data carousel. A top level data carousel contains at least one module known as the BFS directory which includes the module names for all of the other modules on this or any other data carousel. As modules are added to the data carousel, BFS server 12 creates a module name (i.e., identifier) for each new module and then updates the BFS directory structure.

Similarly, when modules are updated and/or deleted from the data carousel, the BFS directory structure is updated by BFS server 12. Applications residing on a multimedia client 14 in turn utilize the BFS directory to access modules contained on the broadcast data carousel. The network 16 may employ any underlying transport protocol (e.g., MPEG transport and/or UDP/IP) which has the ability to deliver data packets across the network to the client.

A broadcast data access system 30 of the present invention serves as the interface between the set of modules contained on the data carousel 32 and the various applications residing 34 on multimedia client 14. Referring to Figure 2, the broadcast data access system 30 on each multimedia client 14 includes an interest manager 36 that is configured to store a plurality of interests. To receive broadcast data from the data carousel 32, an application 34 registers an interest with the interest manager 36 (e.g., a first application registers a first interest requesting a first module from the data carousel). For purpose of this discussion, the term "application" signifies any software module, including the operating system, that may reside on the multimedia client 14. For each module received by the multimedia client 14, a dispatcher 38 accesses the interest manager 36 to determine if any application residing on that client 14 has requested that module, and if so distributes that module to the appropriate application 34.

A more detailed implementation of the broadcast data access system 30 is shown in Figure 3. Preferably, data access system 30 supports at least two data sources: a QAM modulated in-band source and a QPSK modulated out-of-band source. In the case of a set-top box, these multiple data sources may be received via an HFC connection. Since the data access system 30 is designed to support different types of data sources, it provides a data

source processor 42 for each of these data sources. As data blocks arrive at the client, data source processor 42 performs source specific processing on each data block, including but not limited to, removing header information, verifying checksum and decompression.

Once pre-processed by a data source processor 42, data blocks are passed along (in DSM-CC format) to a dispatcher 38. Dispatcher 38 may perform some additional data block verification (e.g., check payload length). However, based on the particular type of DSM-CC message, the dispatcher 38 is primarily responsible for directing each incoming data block to an appropriate processor. Data access system 30 currently incorporates three specific processors: a download information indication (DII) processor 44 which receives DSM-CC download information indication (DII) messages, cancel processor 46 which receives DSM-CC messages that indicate when a module has been removed from the data carousel, and data block processor 48 which receives all other types of data blocks. For the in-band QAM channel, data access system 30 also interacts with a TV manager 52 which provides the functionality to select and manipulate in-band data source.

When an application requests to "open" (or retrieve) a module on the data carousel, it registers this request with the interest manager 36. The interest manager 36 maintains a list of modules which the data access system 30 is currently interested in receiving from the data carousel. Thus, an interest exists for every application request. An identifier (e.g., source id, carousel id, module id, version id, etc.) that uniquely identifies the requested module as well as an identifier for the requesting application are stored in a data structure 37 associated with the interest manager 36. Once an interest has been registered by an application 34, all data blocks that match that interest are processed by the data block processor 48.

In addition to registering each application request, an open file component 36A of the interest manager 36 will interface with the BFS directory 50 to ensure that the requested module exists on the data carousel. Generally, DII messages contain a directory of all of the modules on the carousel and are periodically received (e.g., multiple times per second) by the DII processor 44. As part of this synchronization process, the open file component 36A also verifies that the module exists in accordance to the most recently received DII message. Next, it checks that the version number for the module retrieved from the BFS directory matches the version for that module included in the DII message. In this way, the interest manager 36 ensures that the applications request to open a module is synchronized with the module contained on the data carousel. Because modules are delivered on multiple sources at different rates, it is important to implement this synchronization process; otherwise an application may read data from either an older version of the module or possibly even a newer version of the module. It is also envisioned that a filter may be dynamically installed by the open file component 36A that allows DII messages for this carousel to pass through to the dispatcher 38. Once the synchronization process is complete, the filter is removed so that DII messages will not unnecessarily pass through to the dispatcher 38.

To more efficiently process incoming data blocks, interest manager 36 also dynamically installs filters for each requested interest. Referring to Figure 4, when an application 34 registers an interest, the interest manager 36 installs a filter in the corresponding data source processor 42. A module identifier encapsulated in each data block is compared to each registered interest. Data blocks not associated with any registered interest are discarded before any unnecessary processing occurs on the client. In this way, only data blocks associated with a registered interest undergo pre-processing in the

data source processor 42 and are allowed to pass through to the dispatcher 30. Once all of the data blocks for a particular interest have been received by the client, interest manager 36 removes the filter for that interest.

Typically, a data carousel broadcasts all of the data on the carousel before it re-broadcasts any of the data. Therefore, when an application makes a new data request, the response time depends on the data's location relative to where the carousel is in the broadcast cycle. For example, a carousel having ten data blocks may be currently broadcasting the fifth data block. An application that requests data contained in the seventh block will receive that data faster than an application that requests data contained in the ninth block. If it is the eighth block that is currently being broadcast, the reverse is true: an application that requests data contained in the ninth block will receive that data faster than an application that requests data contained in the seventh block.

Therefore, data block processor 48 reads the data from a data carousel as it is broadcast; it does not wait for the beginning of the data to begin reading. For instance, an application may be interested in a module contained in blocks 2 to 6. If it starts reading as block 4 is broadcast, data block processor 48 copies the data in block 4 to the application's buffer and then reads the data in each successive block that is broadcast. Subsequent data blocks associated with the requested module (i.e., blocks 5, 6, 2, and 3) are copied to the application's buffer, whereas data that is not part of the module (i.e., blocks 7, 8, 9, 10 and 1) are discarded.

Figures 5A and 5B are a flowchart showing a more detailed implementation of the data block processor 48. Start block 100 begins the processing for each incoming data block. In block 102, data block processor 48 interfaces with the interest manager 36 to

retrieve a registered interest. Decision block 104 determines if the incoming data block matches this interest, and if so continues processing in block 106. If the incoming data block does not match the registered interest, data block processor 48 retrieves the next interest. Because all of the interests are evaluated for each data block, multiple interests can be served by the same incoming data block.

Decision block 106 compares the version number for each incoming data block to the expected version number contained in the BFS directory. This is important because a module can be changed on the carousel as individual data blocks for that module are being read by data access system 30. If the version numbers match, then processing proceeds to decision block 110. On the other hand, if the version number does not match, processing branches to an error subroutine 108.

Next, decision block 110 determines if a data block falls within the desired range of data blocks. For each data blocks within the desired range, Block 112 sets a Read Block indicator. Data blocks outside the desired range are discarded in block 114. Read Block indicator is an array data structure used to monitor which data blocks have been read from the data carousel. Each bit in the array represents a data block in the requested module, such that if the bit is on (i.e., set to 1), then the block has been read, but if the bit is off (i.e., set to 0), then the block has not been read. Using this indicator allows data block processor 48 to receive data blocks in any order from the data carousel, and thus eliminates any unnecessary memory copies.

By evaluating the Read Block Indicator, decision block 116 determines if an incoming data block has been previously received by the client. For previously unread data blocks, block 120 sets the appropriate bit in Read Block indicator and then copies the data

block to the corresponding application buffer space (or alternatively placed in cache memory) in block 122. On the other hand, previously read data blocks are discarded in block 118. Decision block 124 evaluates whether all of the data blocks for a requested module have been read (i.e., all bits set to 1). Once all of the data block have been read, data block processor 48 interfaces with interest manager 36 to remove the interest and filter for that requested module in block 126, but in either case processing continues by reading the next interest in block 102. Once all of the interests have been evaluated with respect to an incoming data block, processing is completed for that data block.

Since the data access system 30 receives data from multiple data sources, each of which may have different content as well as different data rates, the BFS directory may not always be in synch with the content of the data carousel being broadcast to the client. For example, a module (including the BFS directory module) can be changed on the carousel as individual data blocks for that module are being read by data access system 30. Therefore, the data block processor 48 also provides a method for downloading and synchronizing the BFS directory on the client.

Figure 6 is a flowchart showing a portion of the data block processor 48 that handles the downloading of the BFS directory. Initially, when a set-top box on an (RF) broadcast network boots, a DSM-CC user-to-network configuration message is received at the client which contains the source ID, carousel ID and module ID for the BFS directory. Based on this information, Block 140 installs a filter that allows (any version of) BFS directory data blocks to pass through the dispatcher 38. As a result, the current version of the BFS directory is retrieved from the data carousel. In a preferred embodiment, BFS directory structure is only delivered over the out-of-band source (which by definition is guaranteed to be available to data access system 30), and thus this filter only need be applied to that data source.

Decision block 144 determines when a BFS directory data block arrives, and if so block 146 reads and parses each incoming data block to construct the BFS directory structure. In the event an error occurs during the downloading process, decision block 148 returns processing to block 142 for re-installation of the initial BFS directory filter. This ensures that the current BFS directory is always downloaded from the data carousel.

Without an error, block 150 checks any registered interests against the newly

downloaded BFS directory. At boot up, there will be no registered interests. However, for subsequently downloaded BFS directories, DII processor 44 interfaces with the interest manager 36 to verify registered interests. If a registered interest no longer exists or has been updated on the data carousel (as indicated by the newly downloaded BFS directory), then the interest is marked for deletion. Block 152 then installs (e.g., cached in memory) the newly downloaded BFS directory onto the client for use by data access system 30.

Once a BFS directory has been successfully downloaded, block 154 installs a different filter which only passes data blocks whose version number does not match the version number of the downloaded BFS directory. When the BFS directory is changed on the server, the version number will be modified, thereby allowing BFS directory data blocks to again pass through the dispatcher 38. Beginning in block 144, a new BFS directory can then be downloaded. In this way, data block processor 48 does not process redundant BFS directory information.

The foregoing discloses and describes merely exemplary embodiments of the present invention. One skilled in the art will readily recognize from such discussion, and from the accompanying drawings and claims, that various changes, modifications and variations can be made therein without departing from the spirit and scope of the present invention.

WHAT IS CLAIMED IS:

1. A broadcast data access system for receiving broadcast data by applications residing on a multimedia client, the broadcast data being a set of modules on a data carousel that are broadcast over a network, comprising:

an interest manager being configured to store a plurality of interests, such that each interest identifies an available module on the data carousel and a requesting application;

a first application on the multimedia client being operative to register a first interest with said interest manager, said first interest representing a first module on the data carousel; and

a dispatcher receiving a plurality of modules from the data carousel, including said first module, and being operative to distribute said first module to said first application by accessing said interest manager.

2. The broadcast data access system of Claim 1 wherein said dispatcher using a module identifier encapsulated in said first module to access said interest manager, thereby determining said first application.

3. A broadcast data access system of Claim 1 wherein said interest manager being operative to remove said first interest after distributing said first module to said first application.

4. The broadcast data access system of Claim 1 further comprises a second application being operative to register a second interest with said interest manager, said second interest representing said first module on the data carousel, and said dispatcher being operative to distribute said first module to said first application and said second application.

5. The broadcast data access system of Claim 1 wherein said first module being

broadcast as a plurality of data packets in a non-sequential order over the data carousel, and said dispatcher being operative to distribute each of said data packets to said first application and to remove said first interest before receiving a duplicate data packet associated with said first module.

6. The broadcast data access system of Claim 1 further comprising a first data source connection between the network and the multimedia client and a second data source connection between the network and the multimedia client, such that said dispatcher receiving said plurality of modules from at least one of said first and second data source connections.

7. The broadcast data access system of Claim 6 further comprising a data source processor connected between each of said data source connections and said dispatcher for performing data source specific processing on each incoming module.

8. The broadcast data access system of Claim 1 further comprising a broadcast file server accessible through the network, said broadcast file server being configured to store said plurality of modules and operative to broadcast said plurality of modules to the multimedia client.

9. A broadcast data access system for receiving broadcast data by applications residing on a multimedia client, the broadcast data being a set of modules on a data carousel that are broadcast over a network, comprising:

a first application on the multimedia client being operative to register a first interest with an interest manager, said first interest representing a first module on the data carousel;

said interest manager being configured to store a plurality of interests, such that each interest identifies an available module on the data carousel and the requesting application;

a data source processor receiving at least a first module and a second module from the data carousel and being configured by said interest manager, in response to said first interest, to filter said second module; and

a dispatcher receiving said first module from said data source processor and distributing said first module to said first application by accessing said interest manager.

10. The broadcast data access system of Claim 9 wherein said dispatcher using a module identifier encapsulated in said first module to access said interest manager, thereby determining said first application.

11. The broadcast data access system of Claim 9 wherein said interest manager being operative to remove said first interest after distributing said first module to said first application.

12. The broadcast data access system of Claim 9 further comprises a second application being operative to register a second interest with said interest manager, said second interest representing said first module on the data carousel, and said dispatcher being operative to distribute said first module to said first application and said second application.

13. The broadcast data access system of Claim 9 wherein said first module being broadcast as a plurality of data packets in a non-sequential order over the data carousel, and said dispatcher being operative to distribute each of said data packets to said first application and to remove said first interest before receiving a duplicate data packet associated with said first module.

14. The broadcast data access system of Claim 9 further comprising a broadcast file server accessible through the network, said broadcast file server being configured to store said plurality of modules and operative to broadcast said plurality of

modules to the multimedia client.

15. A method for accessing broadcast data by applications residing on a multimedia client, the broadcast data being a set of modules on a data carousel that are broadcast over a network, comprising the steps of:

configuring an interest manager to store a plurality of interests, such that each interest identifies an available module on the data carousel and the requesting application;

registering a first interest by a first application with said interest manager, said first interest representing a first module on the data carousel;

installing a filter to pass said first module, in response to said first interest, by said interest manager;

receiving at least a first module and a second module at a data source processor on the multimedia client;

filtering said second module by said data source processor; and

distributing said first module to said first application by accessing said interest manager.

16. The method of Claim 15 wherein the step of distributing said first module further comprises determining said first module based on a module identifier encapsulated in said first module and identifying said first application by using said module identifier to access said interest manager.

17. The method of Claim 15 further comprises the step of removing said filter after distributing said first module to said first application.

18. The method of Claim 15 further comprises the steps of:

registering a second interest by a second application with said interest manager, said

second interest representing said first module on the data carousel; distributing said first module to said second application by accessing said interest manager; and

removing said second interest after distributing said first module to said second application.

5

19. The method of Claim 15 wherein the step of distributing said first module further comprises broadcasting said first module as a plurality of data packets in a non-sequential order over the data carousel and distributing each of said data packets to said first application before receiving a duplicate data packet associated with said first module.

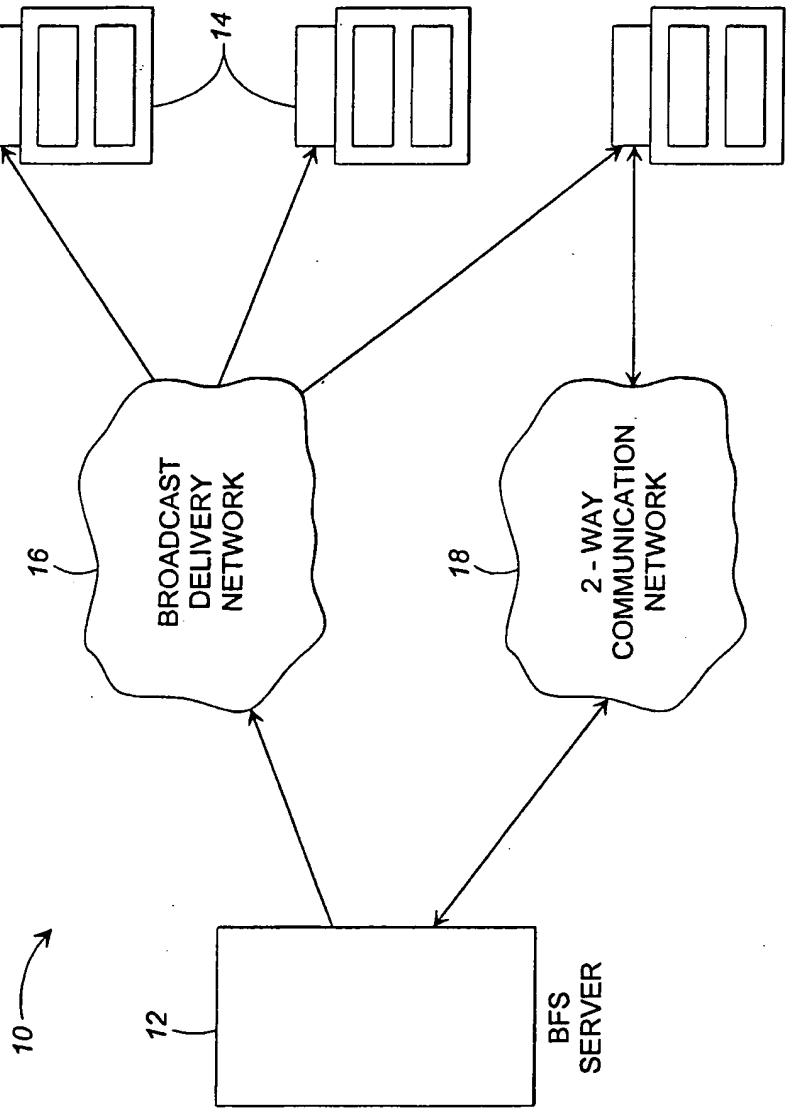

14
BROADCAST DELIVERY NETWORK
16
2 - WAY COMMUNICATION NETWORK
18
BFS SERVER
12
10

FIG. 1

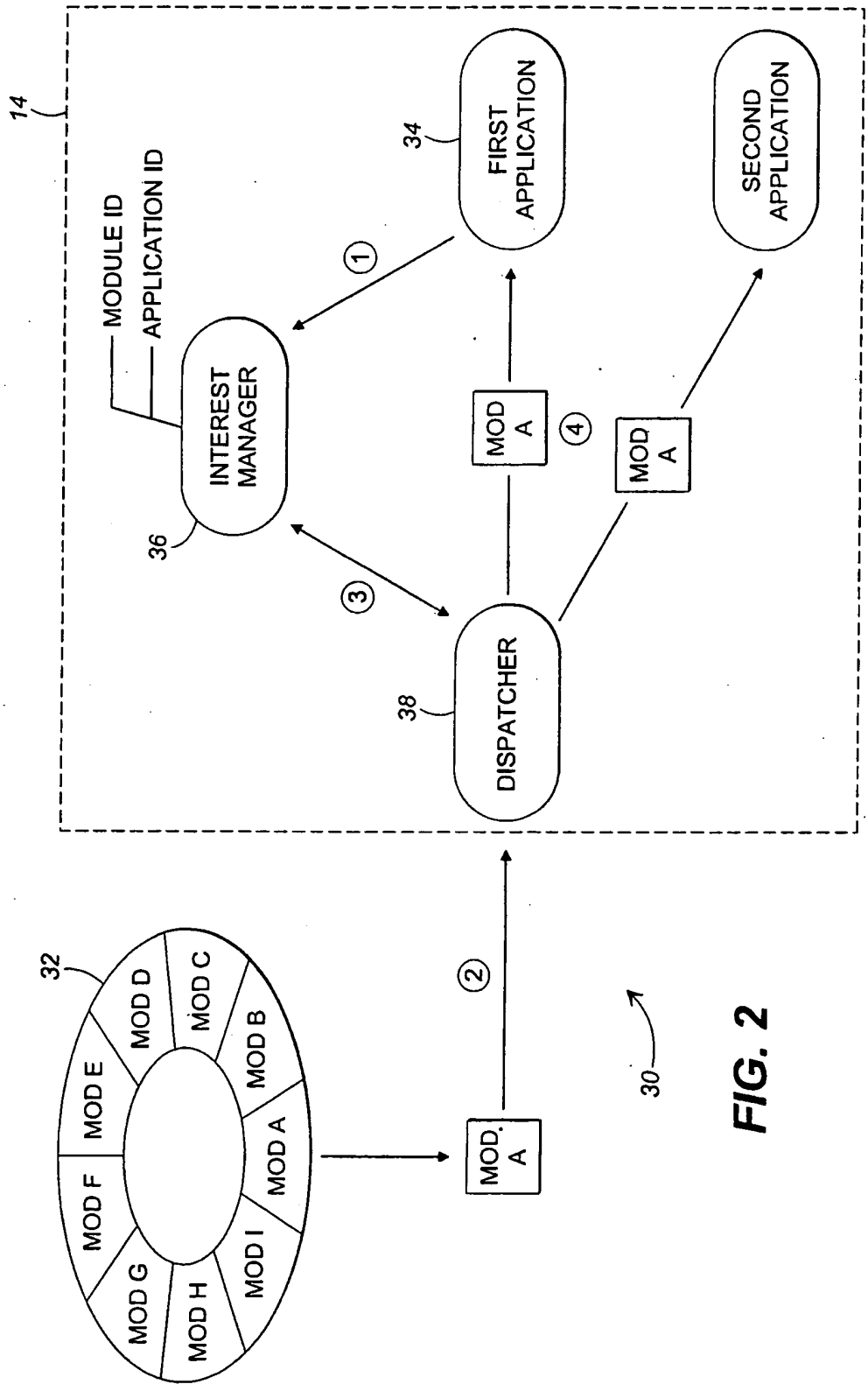

14
MODULE ID
APPLICATION ID
INTEREST MANAGER
36
FIRST APPLICATION
34
SECOND APPLICATION
DISPATCHER
38
MOD A
MOD A
MOD A
MOD. A
32
MOD A
MOD B
MOD C
MOD D
MOD E
MOD F
MOD G
MOD H
MOD I
1
2
3
4
30

FIG. 2

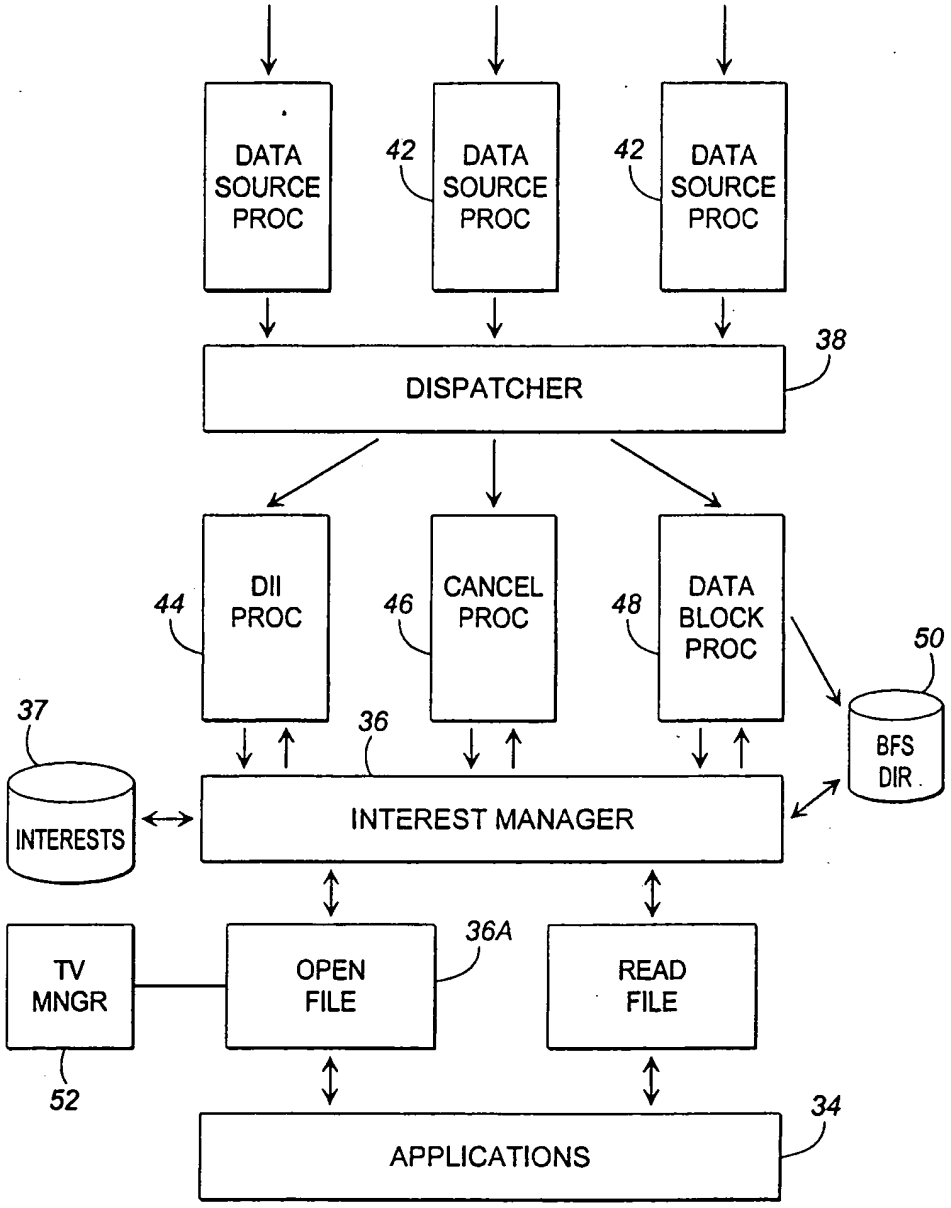

DATA SOURCE PROC
42
DATA SOURCE PROC
42
DATA SOURCE PROC
38
DISPATCHER
44
DII PROC
46
CANCEL PROC
48
DATA BLOCK PROC
50
BFS DIR
37
INTERESTS
36
INTEREST MANAGER
TV MNGR
52
OPEN FILE
36A
READ FILE
34
APPLICATIONS

FIG. 3

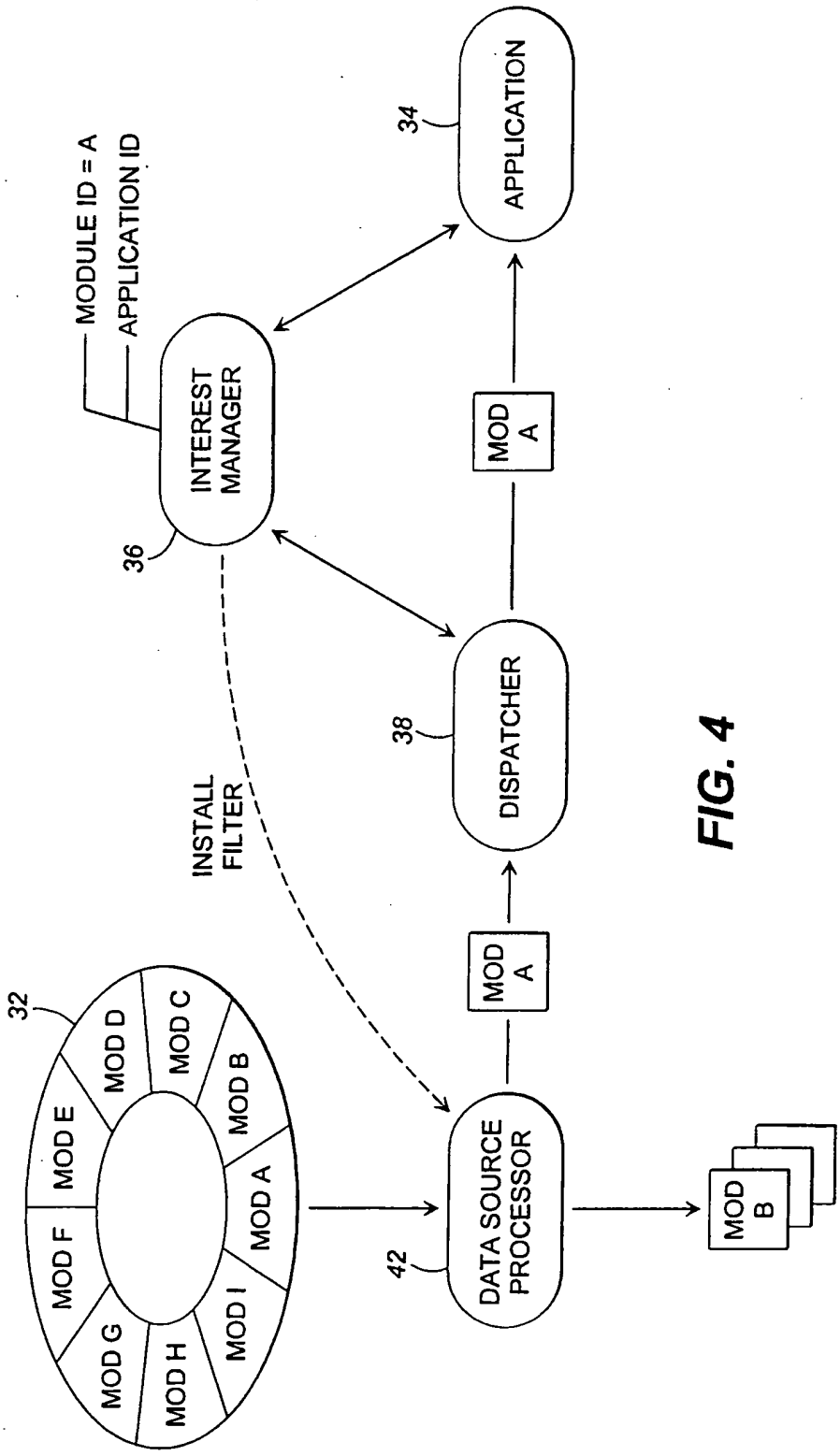

MODULE ID = A
APPLICATION ID
INTEREST MANAGER
36
APPLICATION
34
MOD A
DISPATCHER
38
INSTALL FILTER
MOD A
DATA SOURCE PROCESSOR
42
MOD B
32
MOD A
MOD B
MOD C
MOD D
MOD E
MOD F
MOD G
MOD H
MOD I

FIG. 4

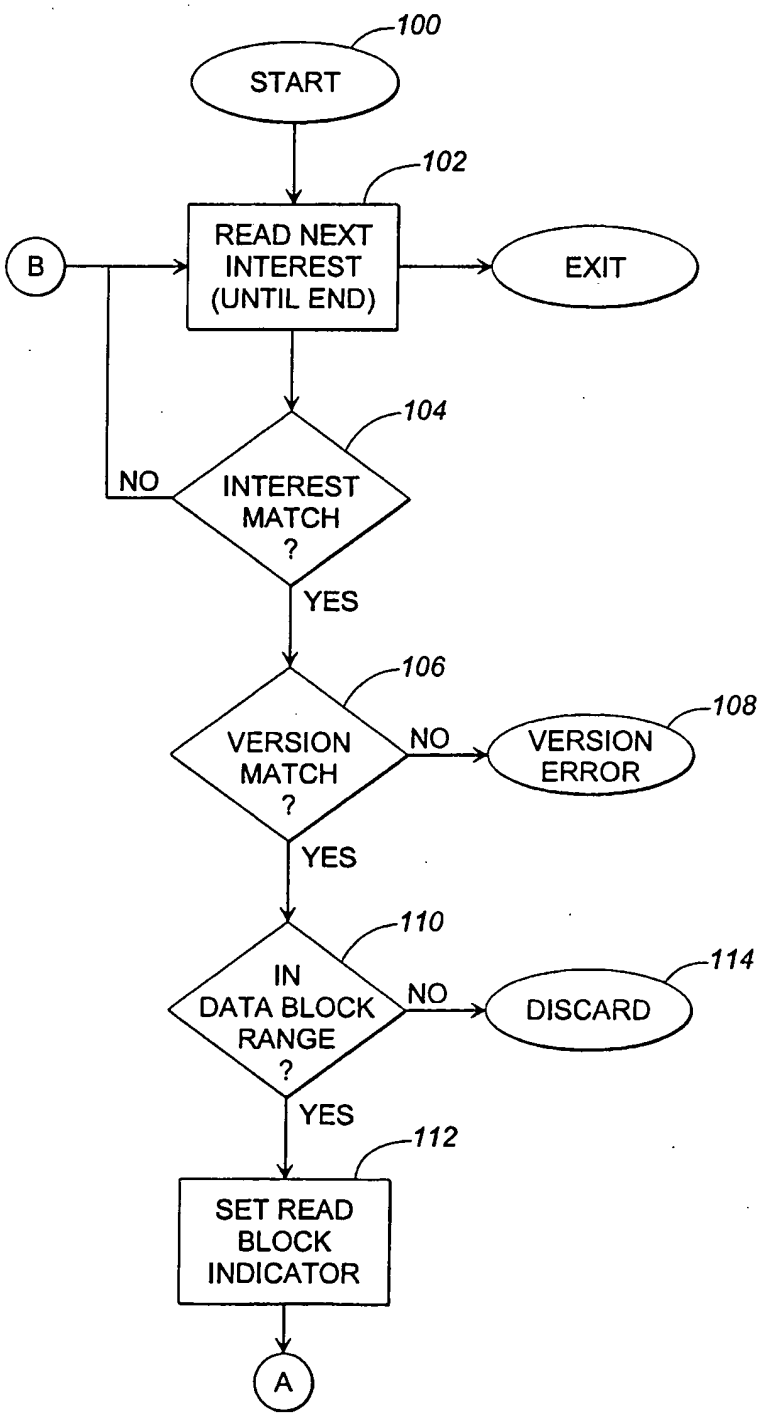

100
START
102
READ NEXT INTEREST (UNTIL END)
B
EXIT
104
NO
INTEREST MATCH ?
YES
106
VERSION MATCH ?
NO
108
VERSION ERROR
YES
110
IN DATA BLOCK RANGE ?
NO
114
DISCARD
YES
112
SET READ BLOCK INDICATOR
A

FIG. 5A

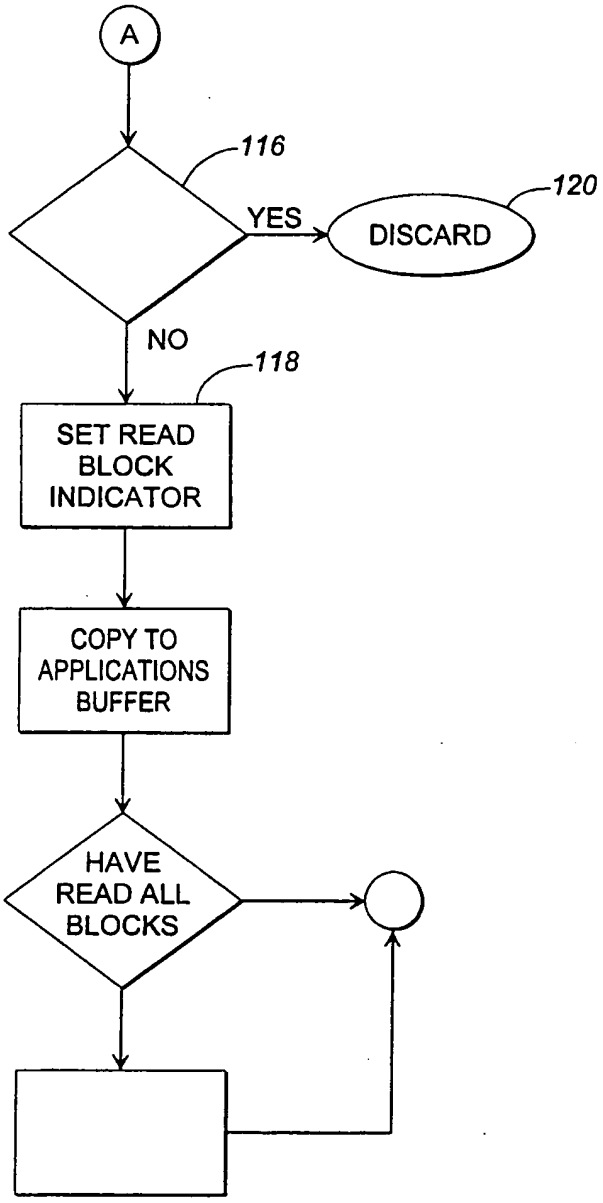

A
116
YES
DISCARD
120
NO
118
SET READ BLOCK INDICATOR
COPY TO APPLICATIONS BUFFER
HAVE READ ALL BLOCKS

FIG. 5B

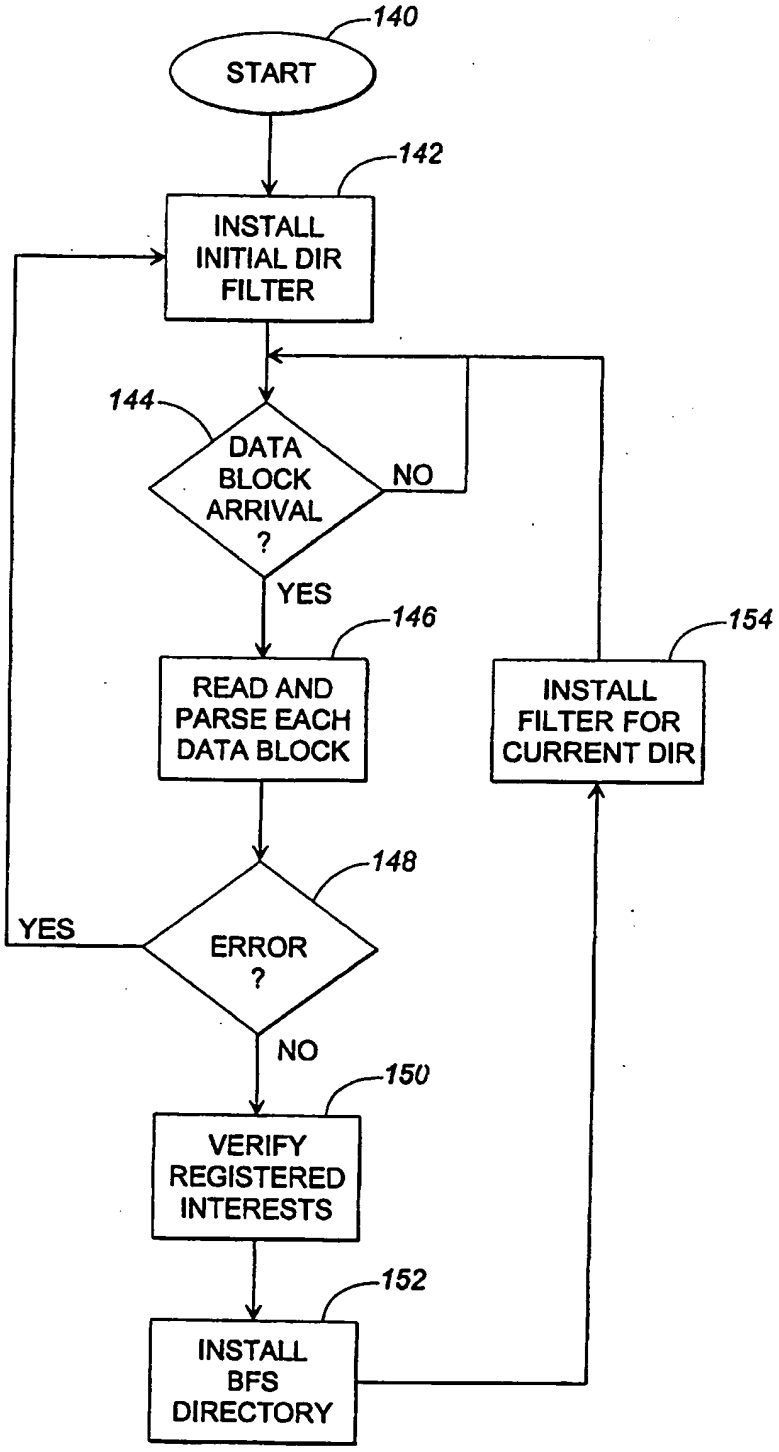

140
START
142
INSTALL INITIAL DIR FILTER
144
DATA BLOCK ARRIVAL ?
NO
YES
146
READ AND PARSE EACH DATA BLOCK
148
ERROR ?
YES
NO
150
VERIFY REGISTERED INTERESTS
152
INSTALL BFS DIRECTORY
154
INSTALL FILTER FOR CURRENT DIR

FIG. 6